(12) UK Patent Application (19) GB (11) 2 375 211 (13) A

(43) Date of A Publication 06.11.2002

(21) Application No **0110810.9**

(22) Date of Filing **02.05.2001**

(71) Applicant(s)
Vox Generation Limited
(Incorporated in the United Kingdom)
Golden Cross House, 8 Duncannon Street, LONDON, WC2N 4JF, United Kingdom

(72) Inventor(s)
Kerry Robinson
David Horowitz

(74) Agent and/or Address for Service
D Young & Co
21 New Fetter Lane, LONDON, EC4A 1DA, United Kingdom

(51) INT CL[7]
G10L 15/06

(52) UK CL (Edition T)
G4R RRL R1F
U1S S2204 S2214

(56) Documents Cited
EP 1089256 A2 **EP 1011094 A1**
EP 0831461 A2 **EP 0241170 A1**
US 6157912 A

(58) Field of Search
UK CL (Edition T) **G4R RRL**
INT CL[7] **G10L 15/00 15/06**
Online:WPI, EPODOC, JAPIO

(54) Abstract Title
Adaptive learning in speech recognition

(57) An adaptive learning unit automatically adapts models of stored utterances in response to use by one or more users. The adaptive learning may be on the basis of recognition hypotheses selected and weighted according to their quality, determined by a hybrid confidence measure, their quantity, and their age.

GB 2 375 211 A

...wing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

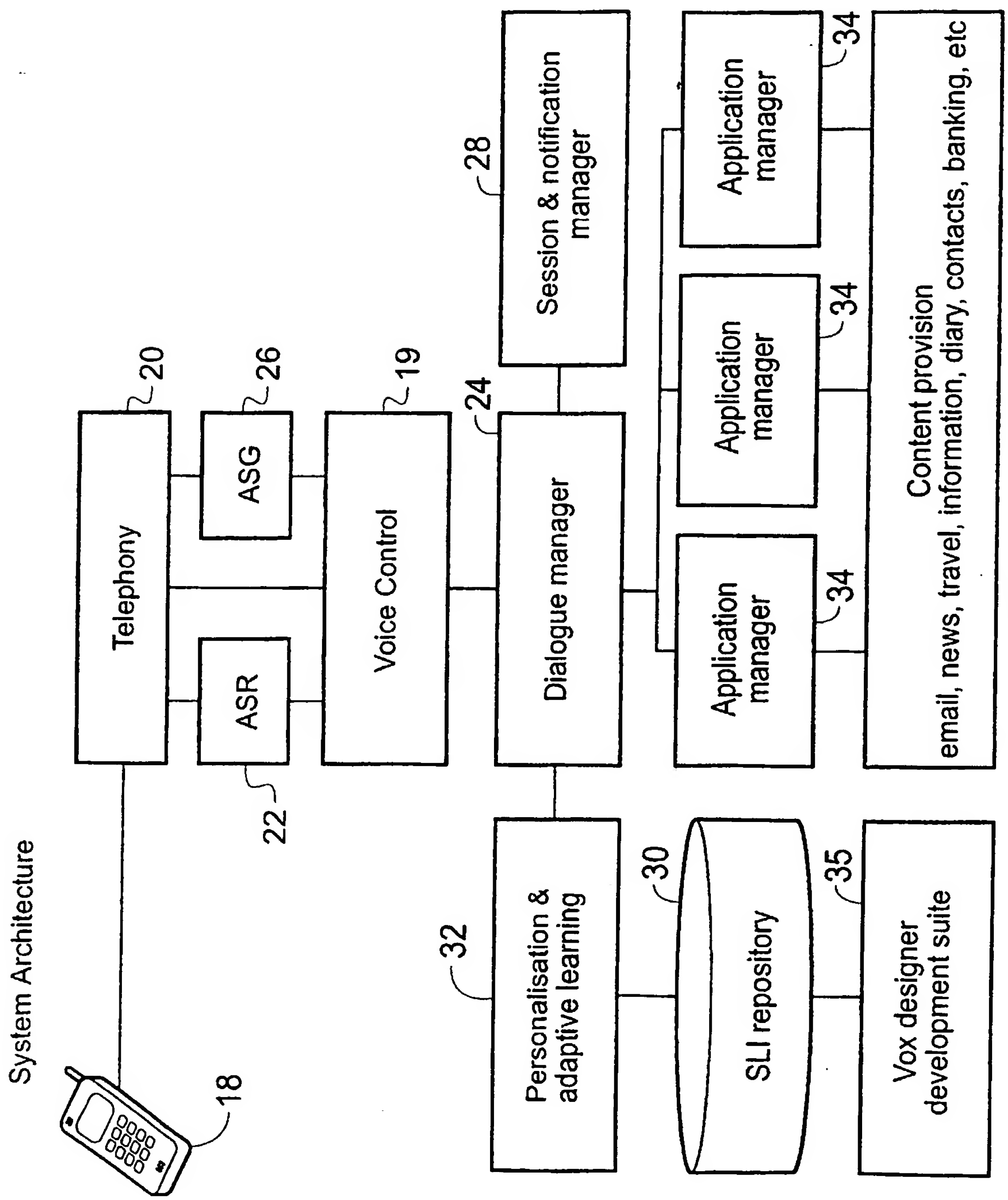
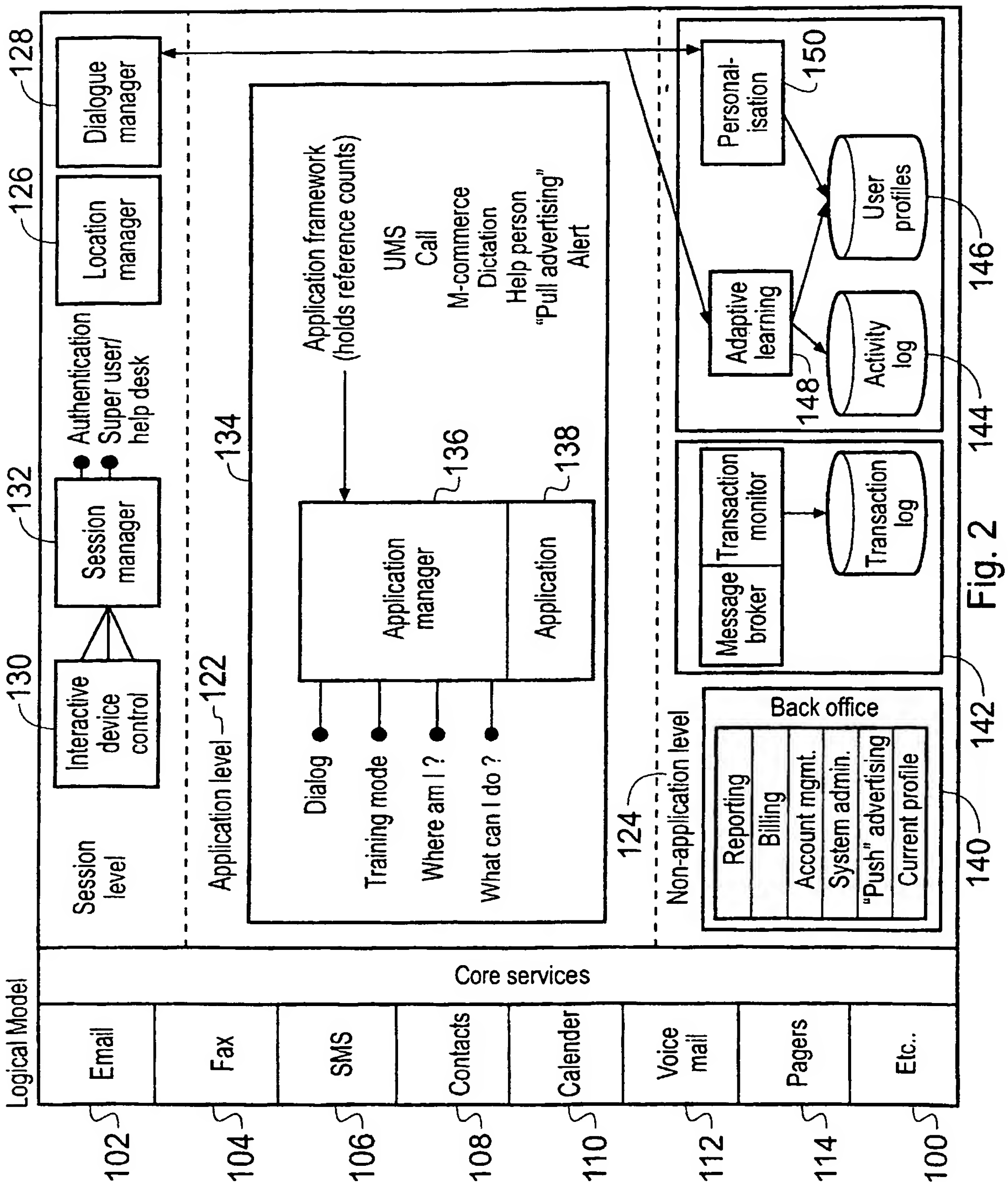
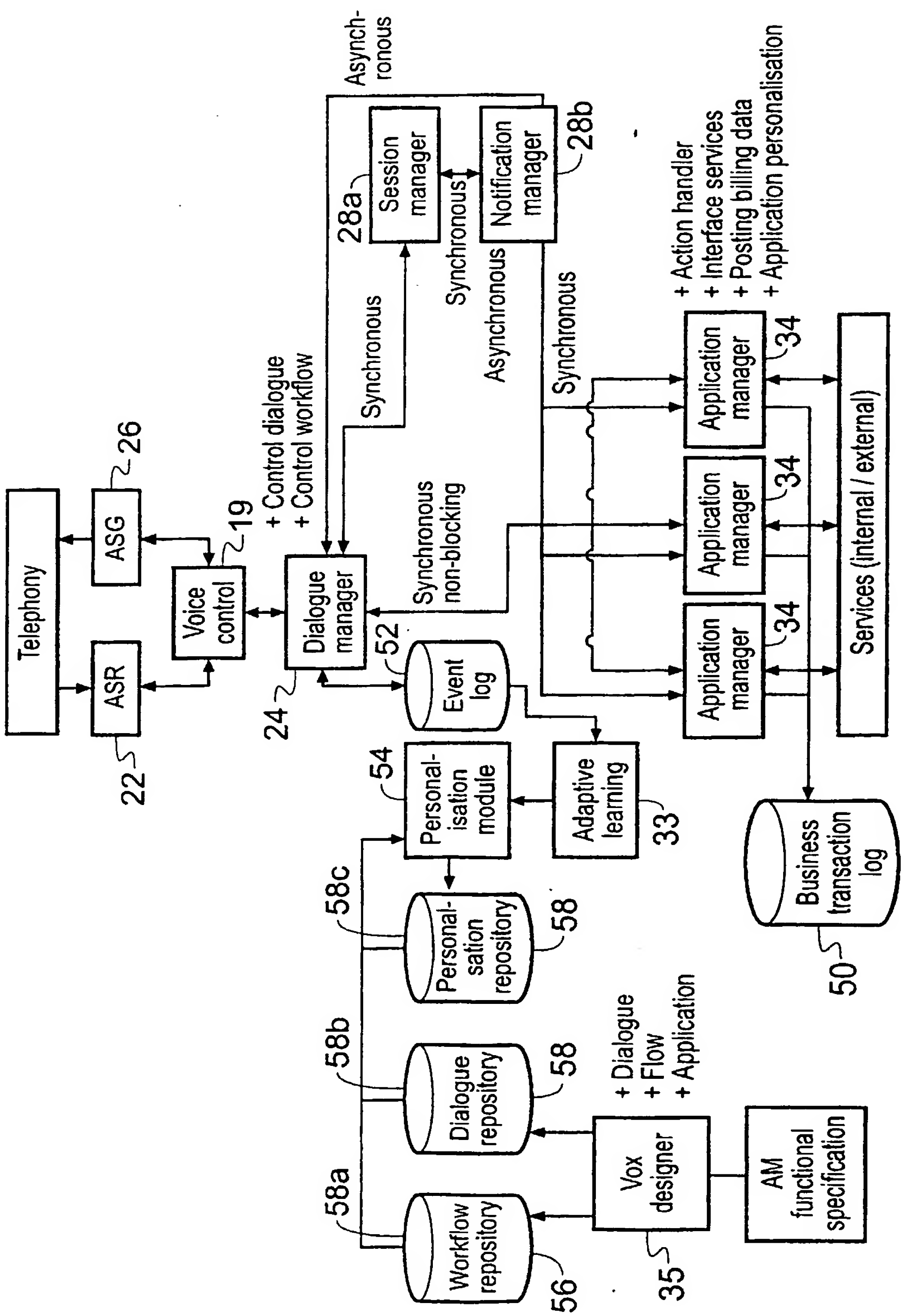

System Architecture
18
Telephony
20
ASR
22
ASG
26
Voice Control
19
Dialogue manager
24
Session & notification manager
28
Application manager
34
Application manager
34
Application manager
34
Content provision
email, news, travel, information, diary, contacts, banking, etc
Personalisation & adaptive learning
32
SLI repository
30
Vox designer development suite
35

Fig. 1



Logical Model
Email
Fax
SMS
Contacts
Calender
Voice mail
Pagers
Etc..
102
104
106
108
110
112
114
100
Core services
Session level
Interactive device control
130
Session manager
132
Authentication
Super user/ help desk
Location manager
126
Dialogue manager
128
Application level
122
134
Application framework (holds reference counts)
UMS
Call
M-commerce
Dictation
Help person
"Pull advertising"
Alert
Application manager
136
Application
138
Dialog
Training mode
Where am I ?
What can I do ?
Non-application level
124
Back office
Reporting
Billing
Account mgmt.
System admin.
"Push" advertising
Current profile
140
Message broker
Transaction monitor
Transaction log
142
Adaptive learning
148
Activity log
144
User profiles
146
Personal-isation
150

Fig. 2



Telephony
ASR
ASG
22
26
Voice control
19
+ Control dialogue
+ Control workflow
Dialogue manager
24
Asynch-ronous
Session manager
28a
Notification manager
28b
Synchronous
Synchronous
Asynchronous
Synchronous
Synchronous non-blocking
+ Action handler
+ Interface services
+ Posting billing data
+ Application personalisation
Application manager
34
Application manager
34
Application manager
34
Services (internal / external)
Event log
52
Personal-isation module
54
Adaptive learning
33
Business transaction log
50
Personal-sation repository
58
58c
Dialogue repository
58
58b
Workflow repository
56
58a
+ Dialogue
+ Flow
+ Application
Vox designer
35
AM functional specification

Fig. 3

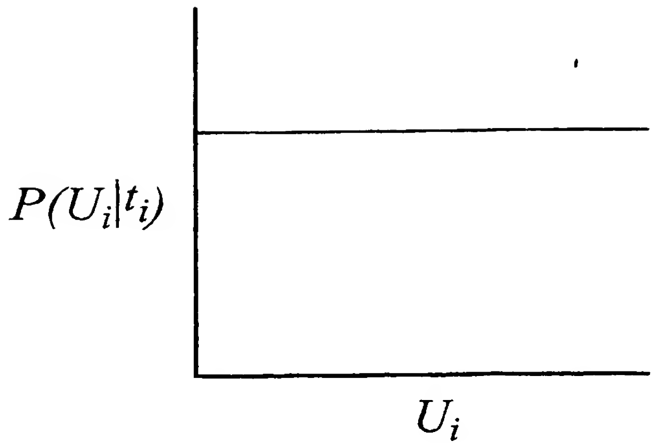


Fig. 4

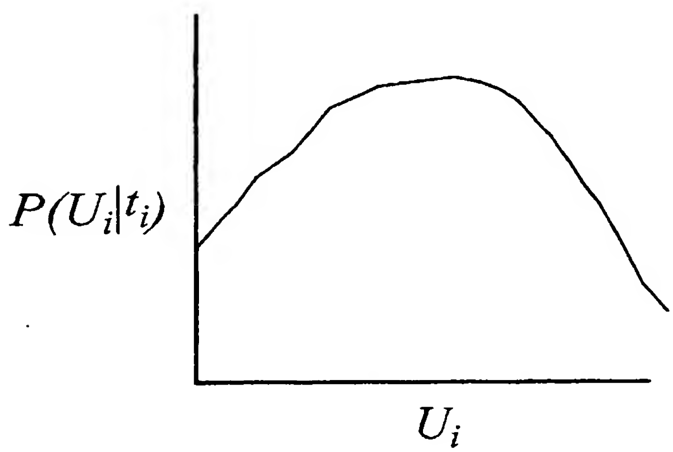


Fig. 5

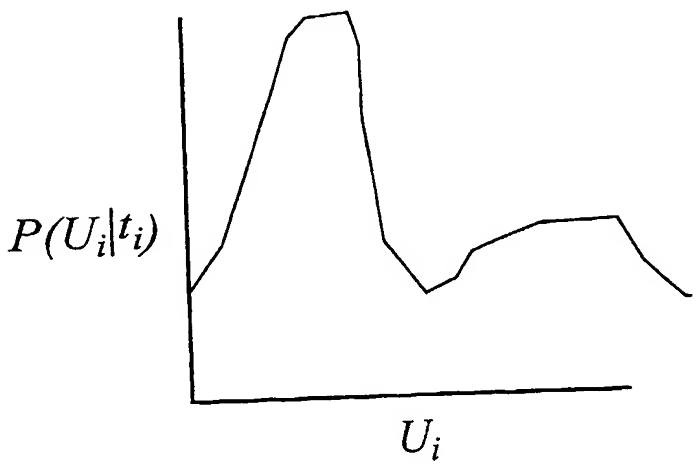


Fig. 6

ADAPTIVE LEARNING IN A PATTERN MATCHING SYSTEM

This invention relates to pattern matching, and in particular to adaptive learning in pattern matching or pattern recognition systems. It is particularly, but not exclusively applicable to adaptive learning of spoken utterances, for example in spoken language interfaces.

Adaptive Learning (AL) is the ability to change certain behaviours as a consequence of historical actions.

In the following description, reference will be made only to spoken language interfaces (SLI). However, it is to be understood that the invention to be described is applicable to any pattern matching systems, including, for example, image recognition.

A spoken language interface (SLI) is a system that allows a user to complete useful tasks by entering into a spoken dialogue with a machine.

Within the context of a spoken language interface (SLI), between a human user and a computer system, there are two places where AL can occur:

1) Human users will adapt their behaviour over time as they become familiar with the expected style and content of an interaction. They will answer questions in the Spoken Language Interface (SLI) with a different choice of phrasing over time. In addition, human users will adapt their behaviour immediately in response to specific events. For example, we might consider that a user adapts to a set of menu options. When they hear the list post-learning they may use barge-in to cut the list off and go straight to the point they want. In addition, the user will learn what is

expected of them in certain places in the dialogue and adapt their behaviour accordingly.

2) The system itself can be trained to adapt as it learns the behaviour patterns of users. It can learn to anticipate the kind of phrasing a user is likely to use and the system should be able to adapt its behaviour in real time - or process during the time the user is not logged on to the system - as a response to specific events. The initially speaker and line independent acoustic models used in the speech pattern matching process can also adapt to accommodate the acoustic peculiarities of a particular speaker or line.

Hence we can view AL in the context of a SLI as an implicit negotiation of communication style between human and machine.

Within a SLI there are several areas in which AL processes may be implemented and accommodated. With regards to human AL, for example, the system can be designed to anticipate gradual user behavioural adaptation. In its simplest form users can be defined as beginners, intermediates, and advanced users and both the quantitative and qualitative nature of prompts can be tailored accordingly. By monitoring the user's experience with the system, prompts are automatically tailored to the task. More verbose help is available for the beginner, terse questions for the experienced user. This definition of users can distinguish between users and can also define progressive behavioural changes within a single user. By detecting progressive behavioural changes, the system can be automatically adapted. Another form of AL concerns changing the prompts and interaction style on the basis of background noise and the strength of a mobile phone signal. The idea here is that the system 'understands' that para-

conversational conditions are changing and adapts its conversational behaviour accordingly. For instance, if the user is using a mobile phone, and there is a large amount of background noise, the system can be trained to predict that it will have difficulty recognising a multi-slot verbose utterance (utterances with many data parameters such as "I want to fly to Paris tomorrow at 3 pm" where Paris, tomorrow and 3 pm are slots. Up to 8 slots may typically be filled. Adapting to these conditions, a message will be played to the user asking them to use single-slot terse speech until conditions improve or by dynamically and automatically limiting the number of slots to a small number (2 - 3). Such system-level adaptive learning is used to guide the user's behaviour and improve recognition accuracy. It does this by automatically adapting the prompt wording. The prompts are stored as phrases and the phrase that is played is predicated on the condition (behaviour or para-conversational).

Typical embodiments of a SLI incorporate components that recognise the natural language of a caller via automatic speech recognition (ASR) and interpret the spoken utterance to allow the caller to complete useful tasks. In order to recognise what the user says, the ASR component must be provided with a model of the language that it expects to hear. The SLI can contain grammars to model language. To allow users to speak naturally, these grammars must specify a large number of utterances to capture the variety of language styles that different users employ: some will say "I want to send an email", others might say "May I send an email" or just "send an email". To accommodate such variability, the number of phrases that the system must listen for at any one time is quite large, but a given caller will not say utterances according to all styles that

have been pre-stored in the SLI grammars. A caller will tend to use certain phraseology and grammatical forms more often than others. Adaptive Learning modifies the language model to selectively listen for utterances that match the style of speech used by callers when talking to the SLI. The adaptation can be made specific to a particular caller, or can be applied to the system as a whole. The result of these existing techniques is a significant decrease in the perplexity of the language model. Perplexity is a measure of the average branching factor of the model - roughly the average number of words that the recogniser must be attempting to match to a segment of the speech signal at any moment in time. Typically, reducing the perplexity of a language model used in ASR will lead to an increase in the recognition accuracy.

There is a need to provide improved recognition accuracy and also to improve the speed of adapting the system.

The invention is defined in the independent claims to which reference should be made.

Preferred features of the invention are defined in the dependent claims to which reference should be made.

Embodiments and preferred embodiments of the invention have the advantage that ASR accuracy problems are solved by automatically modelling and classifying a user's language profile. A user's language style is monitored to selectively tune the recogniser to listen out for the user's preferred utterances. This contrasts to prior art systems which have poor accuracy and use grammars that equally weight a large vocabulary of utterances, or prior art adaptive systems that require human intervention in the adaptation process.

For the avoidance of doubt, the term utterances used herein includes spoken words and speech as well as sounds, abstractions or parts of words or speech.

Embodiments of the present invention will now be described, by way of example, and with reference to the accompanying drawings, in which:

Figure 1 is a schematic view of a spoken language interface;

Figure 2 is a logical model of the SLI architecture;

Figure 3 is a more detailed view of the SLI architecture.

Figure 4 shows a graph of a uniform probability distribution;

Figure 5 shows a graph of a probability of utterance across a language set; and

Figure 6 shows a graph of probability of utterance for a given caller.

The system schematically outlined in Figure 1 is a spoken language interface intended for communication with applications via mobile, satellite, or landline telephone. In the example shown communication is via a mobile telephone 18 but any other voice telecommunications device such as a conventional telephone can be utilised. Calls to the system are handled by a telephony unit 20. Connected to the telephony unit are a Voice Controller 19, an Automatic Speech Recognition System (ASR) 22 and a automatic speech generation system 26. The ASR 22 and ASG systems are each connected to the voice controller 19. A dialogue manager 24 is connected to the voice controller 19 and also to a spoken language interface (SLI) repository 30, a personalisation and adaptive learning unit 32 which is also attached to the SLI repository 30, and a session and notification manager 28. The Dialogue Manager is also connected to a plurality

of Application Managers AM, 34 each of which is connected to an application which may be content provision external to the system. In the example shown, the content layer includes e-mail, news, travel, information, diary, banking etc. The nature of the content provided is not important to the principles of the invention.

The SLI repository is also connected to a development suite 35 that was discussed previously.

Figure 2 provides a more detailed overview of the architecture of the system. The automatic speech generation unit 26 of figure 1 includes a basic TTS unit, a batch TTS unit 120, connected to a prompt cache 124 and an audio player 122. It will be appreciated that instead of using generated speech, pre-recorded speech may be played to the user under the control of the voice control 19. It the embodiment illustrated a mixture of pre-recorded voice and TTS is used.

The system then comprises three levels: session level 120, application level 122 and non-application level 124. The session level comprises a location manager 126 and a dialogue manager 128. The session level also includes an interactive device control 130 and a session manager 132 which includes the functions of user identification and Help Desk.

The application layer comprises the application framework 134 under which an application manager controls an application. Many application managers and applications will be provided, such as UMS (Unified Messaging Service), Call connect & conferencing, e-Commerce, Dictation etc. The non-application level 124 comprises a back office subsystem 140 which includes functions such as reporting, billing, account management, system administration, "push" advertising and current user profile. A transaction

subsystem 142 includes a transaction log together with a transaction monitor and message broker.

In the final subsystem, an activity log 144 and a user profile repository 146 communicate with an adaptive learning unit 148. The adaptive learning unit also communicates with the dialogue manager 128. A personalisation module 150 also communicates with the user profiles repository 146 and the dialogue manager 128.

Referring back to Figure 1, the various functional components are briefly described as follows:

Voice Control 19

This allows the system to be independent of the ASR 22 and TTS 26 by providing an interface to either proprietary or non-proprietary speech recognition, text to speech and telephony components. The TTS may be replaced by, or supplemented by, recorded voice. The voice control also provides for logging and assessing call quality. The voice control will optimise the performance of the ASR.

Spoken Language Interface Repository 30

In contrast to the prior art, grammars, that is constructs and user utterances for which the system listens, prompts and workflow descriptors are stored as data in a database rather than written in time consuming ASR/TTS specific scripts. As a result, multiple languages can be readily supported with greatly reduced development time, a multi-user development environment is facilitated and the database can be updated at anytime to reflect new or updated applications without taking the system down. The data is stored in a notation independent form. The data is converted or compiled between the repository and the voice control to the optimal notation for the ASR being used. This enables the system to be ASR independent.

ASR & ASG (Voice Engine) 22,26

The voice engine is effectively dumb as all control comes from the dialogue manager via the voice control.

Dialogue Manager 24

The dialogue manager controls the dialogue across multiple voice servers and other interactive servers (eg WAP, Web etc). As well as controlling dialogue flow it controls the steps required for a user to complete a task through mixed initiative - by permitting the user to change initiative with respect to specifying a data element (e.g. destination city for travel). The Dialog Manager may support comprehensive mixed initiative, allowing the user to change topic of conversation, across multiple applications while maintaining state representations where the user left off in the many domain specific conversations. Currently, as initiative is changed across two applications, state of conversation is maintained. Within the system, the dialogue manager controls the workflow. It is also able to dynamically weight the users language model by adaptively controlling the probabilities associated with the likely speaking style that the individual user employs dialogue structures in real-time, this is the chief responsibility the Adaptive Learning Engine and the current state of the conversation as a function of the current state of the conversation e user) with the user. The method by which the adaptive learning agent was conceived, is to collect user speaking data from call data records. This data, collected from a large domain of calls (thousands) provides the general profile of language usage across the population of speakers. This profile, or mean language model forms a basis for the first step in adjusting the language model probabilities to improve ASR accuracy. Within a conversation, the individual user's profile is generated and

adaptively tuned across the user's subsequent calls. Early in the process, key linguistic cues are monitored, and based on individual user modelling, the elicitation of a particular language utterance dynamically invokes the modified language model profile tailored to the user, thereby adaptively tuning the user's language model profile and individual increasing the ASR accuracy for that user.

Finally, the dialog manager includes a personalisation engine. Given the user demographics (age, sex, dialect) a specific personality tuned to user characteristics for that user's demographic group is invoked.

The dialog manager also allows dialogue structures and applications to be updated or added without shutting the system down. It enables users to move easily between contexts, for example from flight booking to calendar etc, hang up and resume conversation at any point; specify information either step-by-step or in one complex sentence, cut-in and direct the conversation or pause the conversation temporarily.

Telephony

The telephony component includes the physical telephony interface and the software API that controls it. The physical interface controls inbound and outbound calls, handles conferencing, and other telephony related functionality.

Session and Notification Management 28

The Session Manager initiates and maintains user and application sessions. These are persistent in the event of a voluntary or involuntary disconnection. They can re-instate the call at the position it had reached in the system at any time within a given period, for example 24 hours. A major problem in achieving this level of session

storage and retrieval relates to retrieving a session in which a conversation is stored with either a dialogue structure, workflow structure or an application manager has been upgraded. In the preferred embodiment this problem is overcome through versioning of dialogue structures, workflow structures and application managers. The system maintains a count of active sessions for each version and only retires old versions once the versions count reaches zero. An alternative, which may be implemented, requires new versions of dialogue structures, workflow structures and application managers to supply upgrade agents. These agents are invoked whenever by the session manager whenever it encounters old versions in the stored session. A log is kept by the system of the most recent version number. It may be beneficial to implement a combination of these solutions the former for dialogue structures and workflow structures and the latter for application managers

The notification manager brings events to a user's attention, such as the movement of a share price by a predefined margin. This can be accomplished while the users are offline through interaction with the dialogue manager or offline. Offline notification is achieved either by the system calling the user and initiating an online session or through other media channels, for example, SMS, Pager, fax, email or other device.

Application Managers 34

Application Managers (AM) are components that provide the interface between the SLI and one or more of its content suppliers (i.e. other systems, services or applications). Each application manager (there is one for every content supplier) exposes a set of functions to the dialogue manager to allow business transactions to be realised (e.g. GetEmail(), SendEmail(), BookFlight(),

GetNewsItem(), etc). Functions require the DM to pass the complete set of parameters required to complete the transaction. The AM returns the successful result or an error code to be handled in a predetermined fashion by the DM.

An AM is also responsible for handling some stateful information. For example, User A has been passed the first 5 unread emails. Additionally, it stores information relevant to a current user task. For example, flight booking details. It is able to facilitate user access to secure systems, such as banking, email or other. It can also deal with offline events, such as email arriving while a user is offline or notification from a flight reservation system that a booking has been confirmed. In these instances the AM's role is to pass the information to the Notification Manager.

An AM also exposes functions to other devices or channels, such as web, WAP, etc. This facilitates the multi channel conversation discussed earlier.

AMs are able to communicate with each other to facilitate aggregation of tasks. For example, booking a flight primarily would involve a flight booking AM, but this would directly utilise a Calendar AM in order to enter flight times into a users Calendar.

AMs are discrete components built, for example, as enterprise Java Beans (EJBs) they can be added or updated while the system is live.

Transaction & Message Broker 142 (Fig. 2)

The Transaction and Message Broker records every logical transaction, identifies revenue-generating transactions, routes messages and facilitates system recovery.

<u>Adaptive Learning & Personalisation 32; 148, 150 (Fig.2)</u>

Spoken conversational language reflects quite a bit of a user's psychology, socio-economic background, and dialect and speech style. The reason an SLI is a challenge, which is met by embodiments of the invention, is due to these confounding factors. Embodiments of the invention provide a method of modelling these features and then tuning the system to effectively listen out for the most likely occurring features. Before discussing in detail the complexity of encoding this knowledge, it is noted that a very large vocabulary of phrases encompassing all dialectic and speech style (verbose, terse or declarative) results in a complex listening test for any recogniser. User profiling, in part, solves the problem of recognition accuracy by tuning the recogniser to listen out for only the likely occurring subset of utterance in a large domain of options.

<u>Help Assistant & Interactive Training</u>

The Help Assistant & Interactive Training component allows users to receive real-time interactive assistance and training. The component provides for simultaneous, multi channel conversation (i.e. the user can talk through a voice interface and at the same time see visual representation of their interaction through another device, such as the web).

<u>Databases</u>

The system uses a commercially available database such as Oracle 8I from Oracle Corp.

<u>Central Directory</u>

The Central Directory stores information on users, available applications, available devices, locations of servers and other directory type information.

System Administration - Infrastructure

The System Administration - Applications, provides centralised, web-based functionality to administer the custom build components of the system (e.g. Application Managers, Content Negotiators, etc.),.

Development Suite (35)

This provides an environment for building spoken language systems incorporating dialogue and prompt design, workflow and business process design, version control and system testing. It is also used to manage deployment of system updates and versioning.

Rather than having to labouriously code likely occurring user responses in a cumbersome grammar (e.g. BNF grammar - Bachus Nauer Format) resulting in time consuming detailed syntactic specification, the development suite provides an intuitive hierarchical, graphical display of language, reducing the modelling act to creatively uncover the precise utterance but the coding act to a simple entry of a data string. The development suite enables a Rapid Application Development (RAD) tool that combines language modelling with business process design (workflow).

Dialogue Subsystem

The Dialogue Subsystem manages, controls and provides the interface for human dialogue via speech and sound. Referring to Figure 1, it includes the dialogue manager, spoken language interface repository, session and notification managers, the voice controller 19, the Automatic Speech Recognition Unit 22, the Automatic Speech Generation unit 26 and telephony components 20. The

subsystem is illustrated in more detailed architecture of the interface shown in Figure 3.

Before describing the dialogue subsystem in more detail, it is appropriate first to discuss what is a Spoken Language Interface (SLI).

A SLI refers to the hardware, software and data components that allow users to interact with a computer through spoken language. The term "interface" is particularly apt in the context of voice interaction, since the SLIacts as a conversational mediator, allowing information to be exchanged between user and system via speech. In its idealised form, this interface would be "invisible" and the interaction would, from the user's standpoint, appear as seamless and natural as a conversation with another person. In fact, one principle aim of most SLI projects is to create a system that is as near as possible to a human-human conversation.

If the exchange between user and machine is construed as a dialogue, the objective for the SLI development team is to create the ears, mind and voice of the machine. In computational terms, the ears of the system are created by the Automatic Speech Recognition (ASR) System 22. The voice is created via the Automatic Speech Generation (ASG) software 26, and the mind is made up of the computational power of the hardware and the databases of information contained in the system. The present system uses software developed by other companies for its ASR and ASG. Suitable systems are available from Nuance and Lernout & Hauspie respectively. These systems will not be described further. However, it should be noted that the system allows great flexibility in the selection of these components from different vendors. Additionally, the basic Text To Speech unit supplied, for example, by

Lernout & Hauspie may be supplemented by an audio subsystem which facilitates batch recording of TTS (to reduce system latency and CPU requirements), streaming of audio data from other source (e.g. music, audio news, etc) and playing of audio output from standard digital audio file formats.

One implementation of the system is given in Figure 3. It should be noted that this is a simplified description. A voice controller 19 and the dialogue manager 24 control and manage the dialogue between the system and the end user. The dialogue is dynamically generated at run time from a SLI repository which is managed by a separate component, the development suite.

The ASR unit 22 comprises a plurality of ASR servers. The ASG unit 26 comprises a plurality of speech servers. Both are managed and controlled by the voice controller.

The telephony unit 20 comprises a number of telephony board servers and communicates with the voice controller, the ASR servers and the ASG servers.

Calls from users, shown as mobile phone 18 are handled initially by the telephony server 20 which makes contact with a free voice controller. The voice controller contacts the locates an available ASR resource. The voice controller 19 which identifies the relevant ASR and ASG ports to the telephony server The telephony server can now stream voice data from the user to the ASR server and the ASG stream audio to the telephony server.

The voice controller, having established contacts with the ASR and ASG servers now requests a informs the Dialogue Manager which requests a session on behalf of a user in the session manager. As a security precaution, the user is required to provide authentication information before this step can take place. This request is made to the

session manager 28 which is represented logically at 132 in the session layer in Figure 2. The session manager server 28 checks with a dropped session store (not shown) whether the user has a recently dropped session. A dropped session could be caused by, for example, a user on a mobile entering a tunnel. This facility enables the user to be reconnected to a session without having to start over again.

The dialogue manager 24 communicates with the application managers 34 which in turn communicate with the internal/external services or applications to which the user has access. The application managers each communicate with a business transaction log 50, which records transactions and with the notification manager 28b. Communications from the application manager to the notification manager are asynchronous and communications from the notification manager to the application managers are synchronous. The notification manager also sends communications asynchronously to the dialogue manager 24. The dialogue manager 24 has a synchronous link with the session manager 28a, which has a synchronous link with the notification manager.

The dialogue manager 24 communicates with the adaptive learning unit 33 via an event log 52 which records user activity so that the system can learn from the users interaction. This log also provides a series of debugging and reporting information. The adaptive learning unit is connected to the personalisation module 34 which is in turn connected to the dialogue manager. Workflow 56, Dialogue 58 and Personalisation repositories 60 are also connected to the dialogue manager 24 through the personalisation module 554 so that a personalised view is

always handled by the dialogue manager 24. These three repositories make up the SLI Repository referred to early. As well as receiving data from the workflow, dialogue and personalisation repositories, the personalisation can also write to the personalisation repository 60. The Development Suite 35 is connected to the workflow and dialogue repositories 56, 58 and implements functional specifications of applications storing the relevant grammars, dialogues, workflow and application manager function references for each the application in the repositories. It also facilitates the design and implementation of system, help, navigation and misrecognition grammars, dialogues, workflow and action references in the same repositories.

The dialogue manager 24 provides the following key areas of functionality: the dynamic management of task oriented conversation and dialogue; the management of synchronous conversations across multiple formats; and the management of resources within the dialogue subsystem. Each of these will now be considered in turn.

<u>Dynamic Management of Task Oriented Conversation and Dialogue</u>

The conversation a user has with a system is determined by a set of dialogue and workflow structures, typically one set for each application. The structures store the speech to which the user listens, the keywords for which the ASR listens and the steps required to complete a task (workflow). By analysing what the users say, which is returned by the ASR, and combining this with what the DM knows about the current context of the conversation, based on current state of dialogue structure, workflow structure,

and application & system notifications, the DM determines its next contribution to the conversation or action to be carried out by the AMs. The system allows the user to move between applications or context using either hotword or natural language navigation. The complex issues relating to managing state as the user moves from one application to the next or even between multiple instances of the same application is handled by the DM. This state management allows users to leave an application and return to it at the same point as when they left. This functionality is extended by another component, the session manager, to allow users to leave the system entirely and return to the same point in an application when they log back in - this is discussed more fully later under Session Manager.

The dialogue manager communicates via the voice controller with both the speech engine (ASG) 26 and the voice recognition engine (ASR) 22. The output from the speech generator 26 is voice data from the dialogue structures, which is played back to the user either as dynamic text to speech, as a pre-recorded voice or other stored audio format. The ASR listens for keywords or phrases that the user might say.

Typically, the dialogue structures are predetermined (but stochastic language models could be employed in an implementation of the system or hybrids of the two). Predetermined dialogue structures or grammars are statically generated when the system is inactive. This is acceptable in prior art systems as scripts tended to be simple and did not change often once a system was activated. However, in the present system, the dialogue structures can be complex and may be modified frequently when the system is activated. To cope with this, the dialogue structure is stored as data in a run time repository, together with the

mappings between recognised conversation points and application functionality. The repository is dynamically accessed and modified by multiple sources even when active users are on-line.

The dialogue subsystem comprises a plurality of voice controllers 19 and dialogue managers 24 (shown as a single server in Figure 3).

The ability to update the dialogue and workflow structures dynamically greatly increases the flexibility of the system. In particular, it allows updates of the voice interface and applications without taking the system down; and provides for adaptive learning functionality which enriches the voice experience to the user as the system becomes more responsive and friendly to a user's particular syntax and phraseology with time. Considering each of these two aspects in more detail:

Updates

Today we are accustomed to having access to services 24 hours a day and for mobile professionals this is even more the case given the difference in time zones. This means the system must run none stop 24 hours a day, 7 days a week. Therefore an architecture and system that allows new applications and services or merely improvements in interface design to be added with no affect on the serviceability of the system has a competitive advantage in the market place.

Language Modelling of All SLI System Users.

A typical SLI system works as follows. A prompt is played to a user. The user can reply to the prompt, and be understood, as long as they use an utterance that has been predicted by the grammar writers. Analysis of likely

coverage and usability of the grammars is discussed in another patent application. The accuracy with which the utterance will be recognised is determined, among other things, by the perplexity of the grammars. As large predicted pools of utterances (with correspondingly large perplexity values) are necessary to accommodate a wide and varied user group this may have a negative impact on recognition accuracy. To optimise the balance between coverage and recognition accuracy a frequency based adaptive learning algorithm can automatically assign probability-of-use values to each predicted utterance in a grammar. The algorithm is tuned on an on-going basis by the collection of thousands of utterances pooled across all users. Statistical analysis and summaries of the pooled user data result in the specification of weights applied to the ASR language model. When users use an utterance that has a high probability-of-use the increased probability of the utterance in the grammar increases the probability of correct recognition, regardless of the size of the pool of predicted utterances. Naturally, this method leads to a reduction in recognition accuracy for users who make use of infrequently used utterances that have a low probability-of-use. Modelling of individual users addresses this.

Language Modelling of Individual Users.

Using the same principles described above, AL can be used to probabilistically model the language of individual users. Over the course of repeated uses, the statistics of language use is monitored. Data collection can occur in real-time during the course of the user machine session. Once a model of an individual user's language has been developed it is possible to automatically assign user-

dependent probability-of-use values (by dynamic weighting of the language model) to each predicted utterance in a grammar. As such, user-dependent pools of predicted utterances are automatically identified from within larger pools of predicted utterances. The identification of such sub-pools of utterances, and the assignment of user-dependent probabilities-of-use significantly improve recognition accuracy, especially for very large pools of predicted utterances, where a particular user uses unusual speech patterns.

The AL methodologies described above generalise to novel grammars, such that an experienced user can start to interact with an entirely new application and immediately achieve high levels of recognition accuracy because the language model derived in one application informs the distribution of probabilities in the new application grammars.

A major limitation of AL techniques such as those described above is that they require human transcription of users utterances into text so that they may be used for adaptation.

The following description discusses adapting a language model according to new observations made on-line of the user/caller input utterances as opposed to adaptation using a hand transcribed corpus of caller utterances. Selection of appropriate ASR hypotheses for Real-time automatic adaptation is achieved using a hybrid confidence measure that enables the system itself to decide when sentence hypotheses are sufficiently reliable to be included in the adaptation data. Furthermore, the system can take account of both the quality and quantity of adaptation data when combining language models derived from different training corpora.

Statistical Basis of the Method

The method relies on existing techniques to capture the statistical likelihood of an utterance given the current context of the application. What follows is an outline of some of the existing techniques to which the unique aspects of this invention can be applied.

Consider the probability of a user speaking a certain utterance within a particular context during an interaction with a SLI:

1. *$P(u_i|t_i)$ where u_i is a particular utterance and t_i is the context in which the utterance is spoken.*

The context is some combination of the dialogue state (the current application, the specific user and their interaction history with the system, and the group of users to which the particular user belongs and their collective dialogue histories.

Current [Grammar based] systems of spoken language modelling often assume each utterance in the system occurs with equal likelihood. This is illustrated in Figure 4. However, if one were to observe the occurrences of utterances in the vocabulary set across all users, one would notice that not all utterances are equally weighted. Certain syntactic and semantic forms would be spoken preferentially by the set of callers in comparison to others. One might obtain a probability distribution something like that shown in Figure 5.

If we consider an individual user profile, it is likely that we would observe a different probability function such as that shown in Figure 6.

Current spoken language systems support at most adaptation across a population of users as part of an off-line method to tune the language model. Our work distinguishes itself by being an on-line real-time adaptation methodology that does not depend on human intervention in the adaptation process. The key contribution here is the provision of a method to make individual user observations and early on in the process, adapt a language model specifically to that user profile. Furthermore the adaptation is applied

intelligently according to the quantity and quality of the adaptation data available.

What follows is an example mathematical foundation for this approach. However, the technique is not limited to this particular implementation paradigm.

The example implementation relies on three key components:

1) Using collected user utterances and recognition hypotheses to create a context specific corpus directly or by using this data to select a context specific corpus via information retrieval techniques.

2) Using the context specific corpus to derive language model probabilities.

3) Combining language models intelligently according to the quality (determined by a hybrid confidence measure) and quantities of data from which they are derived.

This approach takes advantage of both rule-based (e.g. Context Free Grammars - CFGs) and data-driven (e.g. n-gram) approaches. The advantage of an n-gram approach is that it quite powerfully models the statistics of language from examples or real usage (the corpus)

. The example sentences used to derive the n-gram probabilities must be representative of language usage in a particular context, but it can be difficult to find enough relevant examples for a new application. The benefit of a CFG approach is that a grammar writer can draw upon their own experience of language to tightly define the appropriate utterances for the topic - but they cannot reliably estimate the likelihood of each of these sentences being used in a live system. Hence there is a need to combine the two approaches, so that grammars may be written to quickly and easily define the scope of a language model in light of the context and data driven approaches can be used to accurately estimate the statistics of real usage.

The following discussion describes a known technique for estimating probabilities form a training corpus.

The grammar writer uses rules to conveniently define a list of possible utterances using an appropriate grammar formalism.

Consider a simple example grammar:

SIMPLE_GRAMMAR([fly travel] to [London Paris])
Covers the following sentences:

I want to fly to London
I want to travel to London
I want to fly to Paris
I want to travel to Paris

We need to estimate the probabilities of each of these sentences using a data driven approach. Ideally we want:

(2) P("<s>I want to fly to London<\s>") which can be estimated by Count("<s>I want to fly to London<\s>") /N

Where:

Count(X) is the number of times sentence X occurs in our training corpus.

N is the number of sentences in our training corpus.

<s> and <\s> are the beginning and end of sentence marker s.

But, to make a reliable estimate of the probability, we need a corpus containing a representative number of examples of the sentence "<s>I want to fly to London<\s>". In practice it is unlikely that we can find such a corpus.

Using Bayes' rule, we can decompose (2)

(3) P("<s>I want to fly to London<\s>") =
P("<\s>" | "<s> I want to fly to London") *
P("London" | "<s> I want to fly to") *
P("to"| "<s> I want to fly") *
P("fly" | "<s> I want to")*
P("to" | "<s> I want") *
P("want" | "<s> I") *
P("I"|<s>)

To estimate these probabilities, we still need many examples of these sentence fragments, some of which are still long. In practice it is usually only possible to find sufficient examples to reliably derive the probabilities of tri-grams - the probability of a particular word following

two others. We assume that the probability of a given word in a sentence is approximately equal to the probability of that word occurring after the previous two words, for example:

P("London"| "<s> I want to fly to") ≈? P("London"| "fly to")

This allows us to write an approximation of P("I want to fly to London") as:

(4) P("<s> I want to fly to London <\s>") ≈?
P("<\s>"| "to London") *
P("London" | "fly to") *
P("to"| "to fly") *
P("fly" | "want to")*
P("to" | "I want") *
P("want" | "<s> I") *
P("I" | "<s>")

We can now expect to be able to estimate the probability of a sentence from the frequency of occurrence of tri-grams in the training data. As an example, consider the following corpus:

I want to go home.

You want to fly to Paris.

I want to fly to London.

We can estimate P("fly" | "want to") as:

P("fly" | "want to")=C("want to fly")/C("want to")

This is simply the number of occurrences of "want to" followed by "fly" in the corpus, divided by the number of occurrences of "want to" (followed by anything).

From the above corpus we find:

C("want to fly")=2

C("want to")=3

So we can estimate:

P("fly" | "want to")=C("want to fly")/C("want to")=3/2

Assigning Probabilities to in Grammars Sentences

The straightforward grammar defines the set of allowable utterances in a domain, but the probability of each utterance is uniform. A grammar constrains the possibilities for the word un that may follow a word history h = u1u2un-1 to a subset of the vocabulary: un={un1, un2, ...unm}. A non-probabilistic grammar assumes that each of these words is equally likely, but

the n-gram approximation described earlier can be used to estimate the probability of each possible word uni following the word history **h**.

P(uni| h) = K* P(uni|un-2,un-1)

We apply the constraint that the probabilities of all of the alternative words must sum to 1 in order to determine the normalisation factor K:

∑?i=1 to m P(uni|h) = K * ∑?i=1 to mP(uni|un-2,un-1) = 1

Rearranging to find K we find:

K= 1/ ∑?i=1 to mP(uni|un-2,un-1)

Substituting the expression (11) for K into (10) we get an expression for the probability of a specific in-grammar word following a given word history **h** = u1u2un-1 in terms of the tri-gram probabilities and the grammar constraint that un={un1, un2, ...unm}:

P(uni| h) = P(uni|u n-2,un-1) / ∑?i=1 to mP(uni|un-2,un-1)

The combination of grammars and statistical language models in this way may be implemented directly in the ASR architecture, or used indirectly to train a probabilistic context free grammar (PCFG), a probabilistic finite state grammar (PFSG) or other probabilistic grammar formalism.

In order for these corpus-derived grammar probabilities to reflect the real usage of sentences in a grammar, the training data must be relevant to it - hence we use an information retrieval measure such as those based on TFIDF in order to select a sub-corpus that is representative of the domain of our grammar.

Term Frequency Inverse Document Frequency (TFIDF)

The TFIDF based similarity measure below is a well known existing technique, described here as an example embodiment of a technique for selecting a subset of a given corpus that most closely relates to a context defined by a query document containing sentences that are:

1. Generated by a grammar.
2. Recognised by the ASR component of a SLI in interaction with a group of users.
3. Recognised by the ASR component of a SLI in interaction with a particular user.

TFIDF vectors can be used as a measure of the similarity of one document (or grammar or set of utterances in a dialogue history) with another. Below is the definition of TFIDF.

TFIDF(for document i , word j) = TF(i,j) * log(IDF(j))

Where:

TF(i,j) is the Term Frequency - the number of times a term j (a word taken from the vocabulary) occurs in document j.

IDF(j) is the inverse document frequency = Total number of documents / Number of documents containing term j.

Each document can be represented as a vector whose components are the TFIDF values for each word in the vocabulary.

To determine the similarity between a query (say, document i=0) and a sentence in a corpus document (document i={1N} where N is the number of documents) we find the cosine of the angle between the TFIDF vector for each document. The cosine of the angle between the two vectors is can be calculated from the dot product of the two vectors:

Cos θ? =M.N / sqrt(N.N)*sqrt(M.M)

Where θ? is the angle between vector N and M. The relevance of this, and other document similarity measures is that we can use a document (or sentences generated from a grammar, or previous dialog history) that we know is specific to our task and compare it with others in a large corpus to find those documents that are similar to the current context.

These documents can then form a sub-corpus that can be used to train the probabilities in an n-gram or probabilistic grammar.

The following is a simple example of the use of TFIDF vectors and the cosine similarity measure in information retrieval:

Consider a corpus:

I want to send an email.

I want to check my bank balance.

I would like to change my PIN number.

Terms (Vocabulary)	IDF (Inverse Document Frequency)
I	3/3
Want	3/2
To	3/2
Send	3/1
An	3/1
Email	3/1
Check	3/1

My	3/2
Bank	3/1
Balance	3/1
Would	3/1
Like	3/1
Change	3/1
PIN	3/1
Number	3/1

TFIDF vector for "I want to send an email" =[1*log(3/3),1*log(3/2),1*log(3/2),1*log(3/1),1*log(3/1),1*log(3/1),0*log(3/1),0*log(3/2),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1)]

=[0,0.176,0.176,0.477,0.477,0.477,0,0,0,0,0,0, 0,0,0]

TFIDF vector for "I want to check my bank balance"

=[1*log(3/3),1*log(3/2),1*log(3/2),0*log(3/1),0*log(3/1),0*log(3/1),1*log(3/1),1*log(3/2),1*log(3/1),1*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1)]

=[0,0.176,0.176,0,0,0,0.477,0.176,0.477,0.477,0,0,0,0,0]

TFIDF vector for "I would like to change my PIN number."

=[1*log(3/3),0*log(3/2),1*log(3/2),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),1*log(3/2),0*log(3/1),0*log(3/1),1*log(3/1),1*log(3/1),1*log(3/1),1*log(3/1),1*log(3/1)]

=[0,0,0.176,0,0,0,0,0.176,0, 0,0.477,0.477,0.477,0.477,0.477]

Now consider a query sentence:

"I want to change my PIN"

TFIDF vector for "I want to change my PIN"

=[1*log(3/3),1*log(3/2),1*log(3/2),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),1*log(3/2),0*log(3/1),0*log(3/1),0*log(3/1),0*log(3/1),1*log(3/1),1*log(3/1),0*log(3/1)]

=[0,0.176,0.176,0,0,0,0,0.176,0, 0,0,0, 0.477,0.477,0]

We now find the similarity between the query sentence and each sentence in the corpus by finding the cosine of the angle between their respective TFIDF vectors:

Similarity("I want to change my PIN", "I want to send an email")=

[0,0.176,0.176,0,0,0,0,0.176,0,0,0,0,0.477, 0.477,0].[0,0.176,0.176,0.477,0.477,0.477,0,0,0,0,0,0,0,0,0]/(SQRT([0,0.176,0.176,0,0,0,0,0.176,0,0,0,0,0.477,0.477,0].[0,0.176,0.176,0,0,0,0,0.176,0,0,0,0,0.477,0.477,0])+ SQRT([0,0.176,0.176,0.477,0.477,0.477,0,0,0, 0,0,0,0,0].[0,0.176,0.176,0.477,0.477,0.477,0,0,0,0,0,0,0,0,0]))

=0.039 (3 s.f.)

Similarity("I want to change my PIN", "I want to check my bank balance")=

[0,0.176,0.176,0,0,0,0,0.176,0,0,0,0,0.477,0.477,0].[0,0.176,0.176,0,0,0,0.477,0.176,0.477,0.477,0,0,0,0,0]/(SQRT([0,0.176,0.176,0,0,0,0,0.176,0,0 ,0,0,0.477,0.477,0].[0,0.176,0.176,0,0,0,0,0.176,0,0,0,0,0.477,0.477,0])+SQRT([0,0.176,0.176,0,0,0,0.477,0.176,0.477,0.477,0,0,0,0,0].[0,0.176,0.176,0,0,0,0.477,0.176,0.477,0.477,0,0, 0,0,0]))

=0.038 (3 s.f.)

Similarity("I want to change my PIN", "I would like to change my PIN number")=

[0,0.176,0.176,0,0,0,0,0.176,0,0,0,0, 0.477,0.477,0].[0,0,0.176,0,0,0,0,0.176,0,0,0.477,0.477,0.477,0.477,0.477]/

(SQRT([0,0.176,0.176,0,0,0,0,0.176,0,0,0,0,0.477,0.477,0].[0,0.176,0.176,0,0,0,0,0.176 ,0,0,0,0,0.477,0.477,0])+SQRT([0,0,0.176,0,0,0,0,0.176,0,0,0.477,0.477,0 .477,0.477,0.477].[0,0,0.176,0,0,0,0,0.176,0,0,0.477,0.477,0.477,0.477,0.477]))

=0.282 (3 s.f.)

All logs are to base 10.

Hence we find that the corpus document "I want to change my PIN" has the greatest similarity with the query "I would like to change my PIN number "

Notice that the inverse document frequency term has caused the similarity measure to ignore the term "I" because it occurs in all corpus documents, and desensitised it for the words "would", " like" because these occur in several documents.

Such a similarity measure can be used to find sub corpora for adaptation:

1) Using a grammar

The grammar is used to generate example sentences. These form the query document.

2) Using On-line adaptation data. Recognition results from users' interaction with the dialogue system are used to form the query document. The query document may contain dialogue histories for a single user or for all users, and these histories may be further divided into application or grammar contexts.

The query document is then compared with all documents in the corpus in order to create a sub corpus containing those documents that are most similar based on a particular measure - such as TFIDF. Hence a sub-corpus is selected that is expected to be more representative of real usage in the context defined by the query document, than the original corpus.

On-line adaptation

The process of speech recognition is highly error prone. Existing AL techniques require users' utterances to be manually transcribed by a human listener before they can be incorporated into a training set for language model adaptation. Embodiments of the present invention enable an automatic on-line adaptation to occur.

We cannot assume that every recognition hypothesis is suitable for use as adaptation data, because the hypotheses may not actually be what the user said. In order to determine which hypotheses are sufficiently accurate to be used in the automatic adaptation process, a hybrid confidence measure is defined to take account of the recogniser confidence values and other accuracy cues in the dialogue history.

A hybrid confidence value is determined for each sentence by a function:

hybrid_c(C,P,S)

Where:

C=Per utterance confidence

P=Preceding dialogue history - e.g. average per-utterance confidence, route to current dialogue state etc.

S=Subsequent dialogue history - e.g. confirms, disconfirms, task switches, asking for help etc.

The form of the function must be determined appropriately for the application and recognition engine.

Then for each utterance we compare it's hybrid confidence measure against a threshold. If the threshold is exceeded then the recogniser's hypothesis for the sentence can be used for automatic adaptation. Otherwise the utterance can be held for manual transcription. If many similar utterances are not correctly recognised because they are

not covered by the grammar, then new rules can be added to the grammars.

Combining sources of Adaption.

The approach outlined above leads to several n-gram language models that must be combined to determine grammar probabilities for a particular user.

We have identified the following statistical language models that may be applied to a grammar:

1) Uniform - essentially the standard CFG produced in the grammar writing process. Each sentence is equally likely.

2) N-gram derived from a sub corpus selected to contain documents similar to sentences covered by the grammar.

3) N-gram derived from all users' dialogue histories, either directly or via selection of a sub corpus.

4) N-gram derived from a particular user's dialogue history, either directly or via selection of a sub corpus.

These models must be combined to create a single n-gram that can be used to define the resulting grammar probabilities. One method of combining them is to use linear interpolation

The probability of a particular word uni following the history h = u1u2…un-1 is:

$$P_{combined}(u_{ni}|h) = \Sigma_j \lambda_j P_j (u_{ni}|h)$$

Where $\Sigma_j \lambda_j = 1$.

Furthermore, the actual interpolation weights can be biased (subject to the condition $\Sigma_j \lambda_j = 1$) according to the hybrid-confidence measures associated with the recognition hypotheses that form the basis of each model, and the total number of examples. Interpolation weights could also be

biased on a per n-gram basis according to the confidences of the individual recognized words that contribute to a particular n-gram, and the hybrid confidence of the sentence in which these words were recognised.

The user adapts to the system, just as the system should attempt to adapt to the user. This means that the user's interaction style is likely to change gradually over time. To accommodate this effect in the system adaptation process, more attention should be paid to more recent dialogue histories than those in the distant past.

The embodiment described enables a spoken user interface (SLI) or other pattern matching system to adapt automatically.

In the case of the SUI, the automatic adaptation is based on grammar probabilities. The embodiment described has the further advantage in that the determination of what data is suitable for use in the adaption process is made according to a hybrid confidence measure. Moreover, individual word confidence scores and the hybrid confidence function can advantageously be used to bias the contribution of individual n-grams in the combined model. Many modifications may be made to the embodiment described without departing from the invention. The principles of the invention are not limited to Automatic Speech Recognition or spoken language interfaces but can be applied to any pattern matching system, including, for example, image recognition.

CLAIMS

1. A Spoken Language Interface for speech communications with an application running on a computer system comprising:

an automatic speech recognition system (ASR) for recognising speech inputs from a user;

a speech generation system for providing speech to be delivered to the user;

wherein the automatic speech recognition system includes a store of predicted utterances and associated probability of use values and compares a speech input from a user with those utterances to recognise the speech input; and further comprising

an adaptive learning unit including an adaptive algorithm for automatically adapting the probabilities associated with the stored utterances in response to use of the Interface by one or more users.

2. A Spoken Language Interface according to claim 1, wherein the automatic adaptation by the adaptive learning unit is made according to recogniser hypotheses.

3. A Spoken Language Interface according to claim 1 or 2, wherein the adaptive learning unit adapts the probabilities associated with the stored utterances on the basis of a hybrid confidence measure.

4. A Spoken Language Interface according to claim 3, comprising a hybrid confidence measure associated with each piece of adaptation data and based on the user dialogue history.

5. A Spoken Language Interface according to claim 4, wherein the hybrid confidence measure is derived from ASR hypothesis confidence values and their related dialogue confirmations and disconfirmations.

6. A Spoken Language Interface according to claim 2, 3 4 or 5, wherein the recognition of speech user inputs by the ASR is based on a language model including a combination of automatically adapted language models based on the quality and quantity of adaptation information.

7. A Spoken Language Interface according to any of claims 1 to 6, wherein the adaptation unit weights the adaptation data according to its age.

8. An adaptive learning unit for a Spoken Language Interface (SLI) for speech communications with an application running on a computer system, the SLI including an automatic speech recognition system (ASR) for recognising speech inputs from a user, a speech generation system for providing speech to be delivered to the user, and a store of predicted utterances for use by the ASR in evaluating utterances from a user to recognise speech input, the adaptive learning unit including an adaptive algorithm for automatically adapting the probabilities associated with the stored utterances in response to use of the Interface by one or more users.

9. An adaptive learning unit according to claim 8, wherein the automatic adaptation by the adaptive learning unit is made according to recogniser hypotheses.

10. An adaptive learning unit according to claim 8 or 9, wherein the probabilities associated with the stored utterances are adapted on the basis of a hybrid confidence measure.

11. An adaptive learning unit according to claim 10, comprising a hybrid confidence measure for each adaptation data item[and is based on the user dialogue historyDELETE, I think the next claim covers this].

12. An adaptive learning unit according to claim 11, wherein the hybrid confidence measure is derived from ASR hypothesis confidence values and their related dialogue confirmations and disconfirmations.

13. An adaptive learning unit according to claim 10, 11 or 12, wherein the recognition of speech user inputs is based on a language model including a combination of automatically adapted language models based on the quality and quantity of adaptation information.

14. An adaptive learning unit according to any of claims 8 to 13, wherein the adaptation unit weights the adaptation data according to its age.

15. An adaptive learning unit for a pattern matching system comprising a store of patterns, the adaptive learning unit comprising an adaptive algorithm for automatically adapting the stored patterns in response to previous pattern matching results.

16. An adaptive learning unit for a pattern matching system according to claim 15, wherein stored patterns are adapted on the basis of previous classification hypotheses, selected and weighted on the basis of a hybrid confidence measure.

17. An adaptive learning unit for a pattern matching system according to claim 15 or 16, wherein the values of past pattern matches used in adapting the stored patterns is weighted according to age.

18. A method of adaptive speech recognition in an automatic speech recognition system (ASR) having a store of predicted utterances, in which speech input from a user is compared with the stored utterances to recognise the input, the method comprising applying an adaptive algorithm to automatically adapt the probabilities associated with the stored utterances in response to use of the Interface by one or more users.

19. A method of adaptive speech recognition according to claim 18, comprising adapting the probabilities associated with stored utterances according to recogniser hypotheses.

20. A method of adaptive speech recognition according to claim 18 or 19, wherein the probabilities associated with the stored utterances are adapted on the basis of a hybrid confidence measure.

21. A method of adaptive speech recognition according to claim 20, wherein a hybrid confidence measure is associated with each item of adaptation information and based on the user dialogue history.

22. A method of adaptive speech recognition according to claim 21, wherein the hybrid confidence measure includes

ASR hypothesis confidence values and their related dialogue confirmations and disconfirmations.

23. A method of adaptive speech recognition according to claim 20,21 or 22, wherein the recognition is based on a language model including a combination of automatically adapted language models determined by the quality and quantity of adaptation information.

24. A method of adaptive speech recognition according to any of claims 19 to 23, comprising weighting the adaptation data according to its age.

25. A method of adaptive learning in a pattern matching system comprising a store of patterns, the method comprising applying an adaptive algorithm to automatically adapt the stored patterns in response to previous pattern matching results.

26. A method of adaptive learning according to claim 25, comprising adapting the stored patterns on the basis of previous match hypotheses, selected and weighted according to a hybrid confidence measure.

27. A method of adaptive learning according to claim 25 or 26, wherein the adaptation data is weighted according to its age.

28. A computer program having program code means which, when run on a computer, cause the computer to perform the method of any of claims 19 to 27.

40

The Patent Office

INVESTOR IN PEOPLE

Application No: GB 0110810.9
Claims searched: 1 to 28

Examiner: John Donaldson
Date of search: 27 February 2002

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.T): G4R(RRL)

Int Cl (Ed.7): G10L 15/00, 15/06

Other: Online: WPI, EPODOC, JAPIO

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
X	EP 1089256 A2	(SONY), see abstract	15 to 17, 25 to 28
X	EP 1011094 A1	(SONY), see abstract	15 to 17, 25 to 28
X	EP 0831461 A2	(NIPPON), see abstract	15 to 17, 25 to 28
X	EP 0241170 A1	(AT&T), see abstract	15 to 17, 25 to 28
X	US 6157912	(KNESER), see abstract, column 1, lines 25 to 31, column 3, line 66 to column 4, line 56	1 to 28

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
		E	Patent document published on or after, but with priority date earlier than, the filing date of this application.
&	Member of the same patent family		